# Hybrid dynamic graph networks for vulnerability detection in blockchain transactions

Hu Xia[1], Toussaint Elvis Compaore[1], Jianbin Gao[1], Qi Xia[1,*], Ansu Badjier[1], Radwa El Bourari[1]
[1]School of Computer Science and Engineering (School of Cyber Security),
University of Electronic Science and Technology of China,
Chengdu, China

***Abstract*—Blockchain technology enables decentralized digital transactions through peer-to-peer exchanges, yet this architecture introduces transaction-level security vulnerabilities exploited by adversaries for fraudulent activities. Existing graph neural network approaches for vulnerability detection exhibit fundamental limitations: static graph analysis neglects temporal transaction evolution, conventional architectures struggle with long-range dependencies, sender-receiver structural asymmetries degrade neighborhood aggregation, and comprehensive adversarial robustness evaluation remains absent. This paper proposes hybrid dynamic graph networks (HDGN), an intelligent system integrating graph attention networks, graph convolutional networks, and bidirectional encoder representations from transformers (BERT) transformers within a unified end-to-end pipeline for robust vulnerability detection in blockchain transaction networks. The framework introduces a missing information prediction module addressing sender-receiver asymmetry through role-aware differential estimation and node-specific affine transformations. A dual vulnerability scoring scheme classifies transactions into four behavioral patterns, enabling fine-grained, interpretable security assessments. Comprehensive adversarial evaluation examines robustness under perturbation edge attack, poisoning attack, and temporal evasion scenarios. Extensive experiments across four blockchain datasets demonstrate HDGN's effectiveness, achieving 93.5% average accuracy (97.7% on Bitcoin Alpha), 95.3% F1-score, and 94.7% AUC while consistently outperforming five state-of-the-art baselines. The framework exhibits strong resilience to structural perturbation attacks with accuracy degradation below 1.0 percentage points. Ablation studies confirm meaningful contributions from each architectural component, with graph attention networks providing the largest performance impact. HDGN advances blockchain security through a robust, interpretable intelligent system demonstrating potential for deployment pending scalability validation.**

***Index Terms*—blockchain security, missing information prediction, vulnerability detection, temporal graph learning, adversarial robustness, fraud detection.**

## I. Introduction

The widespread adoption of blockchain technology has transformed digital transactions, enabling peer-to-peer exchanges without intermediary oversight. However, this decentralization introduces critical security challenges, with fraud losses exceeding billions of dollars annually, particularly in detecting vulnerabilities that malicious actors exploit for fraudulent activities [1]. Blockchain transaction networks exhibit complex, dynamic, and heterogeneous structures where transaction patterns evolve continuously, creating temporal graphs that demand sophisticated intelligent systems for automated analysis [2]. Addressing these challenges requires intelligent systems that leverage structural graph representations and temporal transaction dynamics for comprehensive vulnerability analysis.

Despite advances in graph neural networks (GNNs) for graph-structured data analysis [3], current approaches exhibit five specific limitations when applied to blockchain vulnerability detection. First, existing methods predominantly focus on static graph analysis, neglecting the temporal evolution of transaction networks [4]. Second, conventional GNN architectures struggle to capture long-range dependencies in large-scale transaction graphs [5]. Third, most approaches lack comprehensive adversarial robustness evaluation against perturbation, poisoning, and evasion attacks [6]. Fourth, the fundamental sender-receiver structural asymmetry in blockchain networks remains unaddressed, where receiver nodes often maintain sparse outgoing connections [20]. Finally, severe class imbalance between legitimate and fraudulent transactions poses substantial training challenges [7].

To address the limitations mentioned above, this paper proposes hybrid dynamic graph networks (HDGN), an intelligent system for robust vulnerability detection in blockchain transaction networks and financial systems. The primary contributions of this work are summarized as follows:

- We propose a hybrid sequential architecture integrating graph attention networks, graph convolutional networks, and BERT transformers to capture local attention, global structural propagation, and contextual semantics within a unified end-to-end pipeline.
- To address incomplete data, we design a missing information prediction module that addresses sender-receiver asymmetry through role-aware differential estimation and node-specific affine transformations, reconstructing latent neighborhood information for structurally sparse nodes.
- We also introduce a dual vulnerability scoring scheme that classifies transactions into four behavioral patterns, enabling fine-grained, interpretable security assessments.
- Finally, we conducted a comprehensive adversarial robustness evaluation under perturbation edge attack, poisoning attack, and temporal evasion scenarios, demonstrating exceptional resilience to structural perturbations.

*Corresponding author: Xia Qi (email: xiaqi@uestc.edu.cn).
E-mail: xiahu@uestc.edu.cn (**H. Xia**), compaoreelvis@std.uestc.edu.cn (**T. E. Compaore**), gaojb@uestc.edu.cn (**J. Gao**), 202214080108@std.uestc.edu.cn (**A. Badjie**) and radwa.radwita7@gmail.com (**R. E. Bourari**))

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents preliminaries, problem formulation and feature extraction. Section 4 describes the HDGN framework. Section 5 introduces adversarial attacks. Section 6 present experimental setup, Section 7 report results and analysis and Section 7 concludes with future directions.

## II. Related Work

### A. Traditional and Machine Learning Methods

Traditional network security approaches for blockchain vulnerability detection rely primarily on rule-based expert systems, signature matching, and statistical anomaly detection techniques. Chalapathy and Chawla [8] provide a comprehensive survey of deep learning approaches for anomaly detection, evaluating their effectiveness in handling complex, high-dimensional transaction data. Xia et al. [2] demonstrate LSTM-based fraud detection capabilities, establishing methodological benchmarks for sequential transaction analysis. Xu et al. [7] propose deep boosting decision trees that integrate neural networks into gradient boosting frameworks, improving representation learning while maintaining interpretability. Alarab et al. [9] combine graph convolutional networks with linear layers for anti-money laundering in the Bitcoin blockchain, while Asiri et al. [10] apply graph convolution networks to detect fraudulent transactions in Bitcoin networks. Despite these advances, traditional and classical machine learning approaches exhibit fundamental limitations in capturing the complex relational structures inherent in blockchain transaction networks, motivating the development of graph-based deep learning methods.

### B. Graph Neural Networks for Blockchain Security

Graph neural networks have emerged as powerful tools for modeling relational data in blockchain security. Veličković et al. [3] propose graph attention networks using masked self-attentional layers, enabling adaptive weighting mechanisms for neighborhood aggregation. Hamilton et al. [11] introduce a graph sample and aggregation (GraphSAGE) for inductive node embedding generation through neighborhood sampling. For heterogeneous blockchain networks, Wang et al. [12] propose heterogeneous graph attention networks with hierarchical attention at both node and semantic levels. Applications to blockchain security have demonstrated significant advances. Bilot et al. [13] introduce a phishing detection model (PhishGNN) using Chebyshev spectral convolutions. Cheng et al. [14] propose spatial-temporal attention-based graph networks for credit card fraud detection. For adversarial robustness, Jin et al. [6] develop Pro-GNN for jointly learning clean graph structures and robust GNN parameters, while Chen et al. [15] propose a graph contrastive learning for anomaly detection (GCCAD) using graph contrastive learning to distinguish anomalous nodes based on global context distances.

### C. Temporal Graph Neural Networks

Transaction networks exhibit inherently dynamic properties requiring temporal modeling capabilities. Rossi et al. [16] introduce temporal graph networks with memory modules for processing sequences of timed events. Pareja et al. [4] develop an evolving graph convolutional network (EvolveGCN) to adapt GCN parameters along temporal dimensions using recurrent neural networks. Self-attention mechanisms have proven effective for temporal analysis. Sankar et al. [17] introduce dynamic self-attention networks with joint attention along structural and temporal dimensions. Tian and Liu [19] propose spatial-temporal-aware graph transformers employing temporal encoding strategies with pairwise node interactions. However, existing patterns, as evidenced by performance gaps exceeding 10 percentage points on standard benchmarks [4], [17], [18], do not comprehensively address adversarial robustness or sender-receiver structural asymmetry. The proposed HDGN addresses these gaps through its hybrid GAT-GCN-BERT architecture, missing information prediction module, and comprehensive robustness evaluation.

## III. Preliminaries, Problem Statement & Feature Extraction

This section establishes preliminaries, including formal definitions, problem formulation, and feature extraction, underlying the HDGN framework for blockchain vulnerability detection.

### A. Preliminaries

*Definition 1 (Temporal Graph Snapshot):* A temporal graph snapshot is formally defined as $G_t = (V_t, E_t, X_t, A_t)$, where $V_t$ denotes the set of blockchain addresses active during time window $t$, $E_t \subseteq V_t \times V_t$ represents directed transaction edges, $X_t \in \mathbb{R}^{|V_t| \times d}$ is the node feature matrix with $d$-dimensional feature vectors, and $A_t \in \{0,1\}^{|V_t| \times |V_t|}$ is the adjacency matrix encoding transaction connectivity.

The temporal graph construction employs sliding windows of $\Delta t = 14$ days with a stride $\tau = 2$ days, generating overlapping snapshots that capture evolving transaction patterns across time.

*Definition 2 (Ego Network Subgraph):* For a center node $v$ in graph $G_t$, the $k$-hop ego network $G_v^{(k)}$ is the induced subgraph containing all nodes within $k$ hops from $v$ and their interconnecting edges. The ego network captures local structural patterns around individual addresses, enabling efficient computation and focused vulnerability assessment. We set $k = 2$ based on preliminary experiments showing diminishing returns beyond 2-hop neighborhoods and prior work demonstrating 2-hop sufficiency for fraud detection [14].

*Definition 3 (Dual Vulnerability Scores):* For a node $v$ in temporal graph $G_t$, the vulnerability score $\psi_v^+$ quantifies the propensity to initiate or participate in malicious transactions, while the anti-vulnerability score $\psi_v^-$ measures resilience against fraudulent activities. These scores are computed as weighted aggregations: $\psi_v^+ = \alpha \cdot s_{\text{behav}}(v) + \beta \cdot s_{\text{struct}}(v) + \gamma \cdot s_{\text{temp}}(v)$, where $s_{\text{behav}}$, $s_{\text{struct}}$, and $s_{\text{temp}}$ represent behavioral,

structural, and temporal indicators respectively, with weights $\alpha = 0.4$, $\beta = 0.35$, $\gamma = 0.25$ determined through sensitivity analysis.

*Definition 4 (Behavioral Patterns):* Behavioral patterns characterize node transaction activities through four categories derived from vulnerability score combinations: Normal-Normal (NN) where both sender and receiver exhibit normal behavior ($\psi_s^+ \leq \theta \wedge \psi_r^+ \leq \theta$), Normal-Abnormal (NA) with normal sender and abnormal receiver, Abnormal-Normal (AN) with abnormal sender and normal receiver, and Abnormal-Abnormal (AA) where both endpoints exhibit abnormal behavior, where $\theta = 0.5$ denotes the vulnerability threshold. These patterns provide post-hoc interpretability for security analysts prioritizing edge types during investigation.

### B. Problem Statement

Given a blockchain transaction network represented as a temporal dynamic graph $\mathcal{G} = \{G_1, G_2, \ldots, G_T\}$ where $G_t = (V_t, E_t, X_t, A_t)$ denotes the network state at timestamp $t$, the objective is to learn a mapping function $f : E_t \to \{0, 1\}$ that classifies each edge $e_{uv} \in E_t$ as vulnerable (1) or non-vulnerable (0). This problem presents three fundamental challenges: (1) severe class imbalance with vulnerable edges of transactions, requiring specialized loss functions and sampling strategies; (2) sender-receiver structural asymmetry where transaction initiators exhibit different neighborhood characteristics than recipients, necessitating role-aware feature learning; and (3) temporal evolution of attack patterns requiring models that capture both instantaneous graph topology and longitudinal behavioral dynamics. The intelligent system must achieve high precision to minimize false alarms while maintaining high recall to detect actual threats, operating under real-time detection constraints for production blockchain monitoring.

### C. Feature Extraction

The feature extraction module transforms ego network subgraphs into comprehensive numerical representations capturing structural, temporal, and behavioral characteristics. Algorithm 1 operates on ego network $G_v^{(k)}$ centered at node $v$ with timestamps $\mathcal{T}_v$, producing feature vector $\mathbf{f}_v \in \mathbb{R}^{14}$.

**Algorithm 1** Feature Extraction from Ego Networks

**Require:** Ego network subgraph $G_v^{(k)} = (V_{\text{ego}}, E_{\text{ego}})$, center node $v$, timestamps $\mathcal{T}_v$
**Ensure:** Feature vector $\mathbf{f}_v \in \mathbb{R}^{14}$
**Phase I: Structural Features**
Compute degrees: $d_{\text{in}}(v), d_{\text{out}}(v), r_d(v) \leftarrow d_{\text{out}}/\max(d_{\text{in}}, 1)$
Compute volumes: $\text{vol}_{\text{in}}(v), \text{vol}_{\text{out}}(v), \delta_{\text{vol}}(v)$
Compute topology: $\text{CC}(v), \rho(G_v^{(k)}), \text{PR}(v), \text{BC}(v)$
**Phase II: Temporal Features**
Compute activity metrics: $\text{span}(v), \text{freq}(v), \text{burst}(v), \text{recency}(v)$
**Phase III: Aggregation**
$\mathbf{f}_v \leftarrow [d_{\text{in}}, d_{\text{out}}, r_d, \text{vol}_{\text{in}}, \text{vol}_{\text{out}}, \delta_{\text{vol}}, \text{span}, \text{freq}, \text{burst}, \text{recency}, \text{CC}, \rho, \text{PR}, \text{BC}]$
Normalize: $\mathbf{f}_v \leftarrow (\mathbf{f}_v - \boldsymbol{\mu})/\boldsymbol{\sigma}$
**return** $\mathbf{f}_v$

Phase I extracts structural features: in-degree $d_{\text{in}}(v)$ counts incoming transactions, out-degree $d_{\text{out}}(v)$ counts outgoing transactions, degree ratio $r_d(v)$ quantifies sender-receiver imbalance, incoming volume $\text{vol}_{\text{in}}(v)$ and outgoing volume $\text{vol}_{\text{out}}(v)$ measure transaction amounts, volume difference $\delta_{\text{vol}}(v)$ captures imbalance, clustering coefficient $\text{CC}(v)$ measures local connectivity, density $\rho(G_v^{(k)})$ quantifies ego network connectivity, PageRank $\text{PR}(v)$ indicates importance, and betweenness centrality $\text{BC}(v)$ measures bridging position. Phase II computes temporal features: activity span $\text{span}(v)$ measures temporal range, frequency $\text{freq}(v)$ quantifies transaction rate, burstiness $\text{burst}(v)$ captures timing irregularity, and recency $\text{recency}(v)$ indicates elapsed time since last activity. Phase III concatenates features into a 14-dimensional vector and applies z-score normalization $(\mathbf{f}_v - \boldsymbol{\mu})/\boldsymbol{\sigma}$ for numerical stability during neural network processing, where $\mu$ and $\sigma$ denote the per-feature mean and standard deviation computed over the training set.

## IV. Proposed Method: HDGN Framework

This section presents the hybrid dynamic graph networks (HDGN) framework for vulnerability detection in blockchain transaction networks.

### A. Architecture Overview

The HDGN framework, illustrated in Fig. 1, processes normalized 14-dimensional feature vectors $\mathbf{f}_v$ from ego network extraction as input. The framework performs binary edge-level vulnerability detection, classifying transactions as vulnerable (security threats) or non-vulnerable (legitimate activity). Sequential processing includes: (1) Graph attention networks for adaptive local aggregation, (2) missing information prediction addressing sender-receiver asymmetry, (3) Graph convolutional networks for global pattern propagation, (4) BERT transformers for contextualized edge representations, and (5) MLP-based vulnerability classification producing final predictions.

### B. Graph Attention Network with Adaptive Aggregation

The first neural processing stage employs graph attention networks (GAT) for adaptive neighborhood aggregation, focusing on structurally relevant neighbors while downweighting noisy connections. For each node $v$ at layer $l$, GAT computes attention coefficients $\alpha_{v,u}$ quantifying the importance of neighbor $u$ in message passing:

$$\alpha_{v,u} = \frac{\exp(e_{v,u})}{\sum_{k \in \mathcal{N}(v)} \exp(e_{v,k})} \tag{1}$$

where the unnormalized attention score is:

$$e_{v,u} = \text{LeakyReLU}\left(\boldsymbol{\alpha}^\top \left[\mathbf{W}\mathbf{h}_v^{(l-1)} \| \mathbf{W}\mathbf{h}_u^{(l-1)}\right]\right) \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{d' \times d}$ represents the learnable weight matrix, $\alpha \in \mathbb{R}^{2d'}$ denotes attention parameters, $\|$ indicates concatenation, and $\mathcal{N}(v)$ defines node $v$'s neighborhood. The LeakyReLU activation with a negative slope of 0.2 prevents gradient saturation during training.

Node representations are updated through attention-weighted aggregation:

$$\mathbf{h}_v^{(l)} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{v,u} \mathbf{W}\mathbf{h}_u^{(l-1)}\right) \tag{3}$$

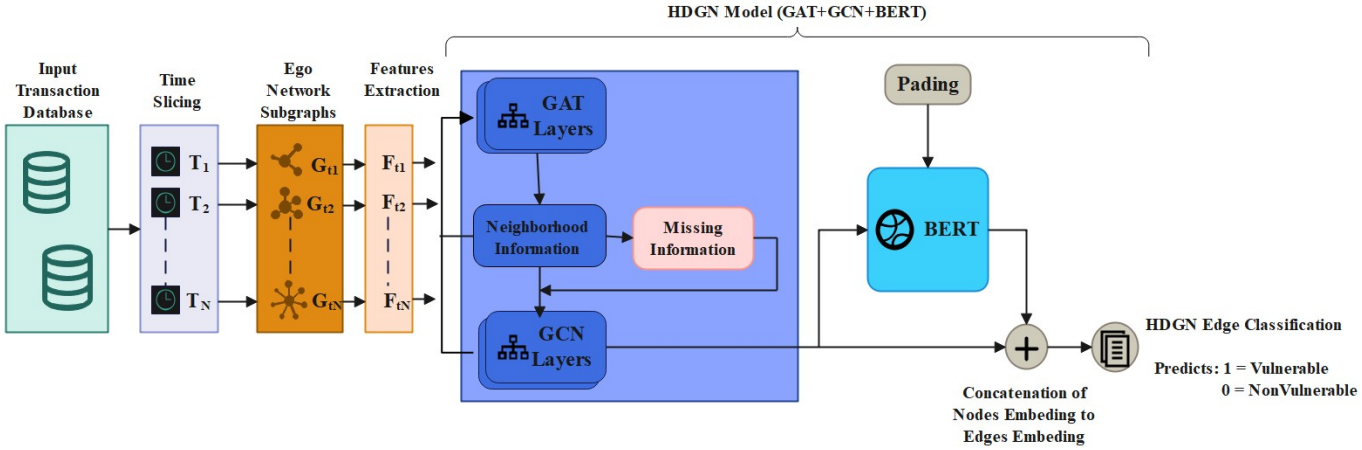


Fig. 1. Architecture of the Hybrid Dynamic Graph Networks (HDGN) framework.

where $\sigma$ denotes the RELU activation function. Multi-head attention with $K = 4$ heads captures diverse structural patterns, with head outputs concatenated and linearly transformed. This attention mechanism is essential for fraud detection because it enables the model to distinguish between legitimate high-volume traders and fraudulent accounts exhibiting similar connectivity patterns.

### C. Missing Information Prediction Module

Blockchain transaction networks exhibit structural asymmetries from directional transaction flows. Sender nodes develop rich outgoing neighborhoods, while receiver nodes maintain sparse connections, particularly for merchant accounts and exchanges. This asymmetry creates incomplete representations that degrade message passing effectiveness.

The module addresses this by estimating ideal neighborhood embeddings incorporating observed neighbors and latent relationships. Following tail-node handling frameworks [20], neighborhood embeddings after GAT processing are approximated through additive decomposition:

$$\mathbf{h}_{N_v}^t \approx \mathbf{h}_v^t + \mathbf{r}_v^t \tag{4}$$

where $\mathbf{h}_{N_v}^t \in \mathbb{R}^d$ represents the aggregated neighborhood embedding, $\mathbf{h}_v^t$ denotes the central node embedding at snapshot $t$, and $\mathbf{r}_v^t = \text{MLP}_r(\mathbf{h}_v^t)$ is predicted by a two-layer MLP capturing information flow patterns.

The missing neighborhood information $\mathbf{m}_v^t$ quantifies the discrepancy between ideal and observed representations:

$$\mathbf{m}_v^t = \mathbf{h}_{N_v^*}^t - \mathbf{h}_{N_v}^t \tag{5}$$

where $N_v^*$ denotes the ideal neighborhood and $N_v$ the observed neighborhood. The ideal embedding $\mathbf{h}_{N_v^*}^t$ is approximated using neighborhoods from high-degree nodes with similar structural roles via degree and PageRank similarity. The module personalizes a globally shared vector $\mathbf{r}^t$ through learned affine transformation:

$$\mathbf{r}_v^t = (\boldsymbol{\gamma}_v^t + \mathbf{1}) \odot \mathbf{r}^t + \boldsymbol{\beta}_v^t \tag{6}$$

where $\boldsymbol{\gamma}_v^t, \boldsymbol{\beta}_v^t \in \mathbb{R}^d$ denote element-wise scaling and shifting vectors through learned projections, $d = 256$ is the hidden dimension (constant across all snapshots), and $\odot$ indicates Hadamard product. The enhanced GAT output is:

$$\mathbf{H}_{gat}^{(t+1)} = \text{GAT}(\alpha, \mathbf{f}^t, \mathbf{H}^{(t)}, \mathbf{W}) + \lambda \mathbf{M}^t \tag{7}$$

$$\mathbf{H} = \mathbf{H}_{gat}^{(t+1)} \tag{8}$$

where $\mathbf{M}^t$ represents predicted missing information and $\lambda = 0.3$ controls contribution weight.

### D. Graph Convolutional Network Processing

Following attention-based aggregation and missing information prediction, HDGN employs graph convolutional networks to capture global structural patterns through spectral convolution. While GAT focuses on learning local neighborhood importance, GCN propagates information across the entire graph structure, enabling the detection of patterns spanning multiple transaction hops.

The normalized adjacency matrix is constructed as:

$$\hat{A} = \tilde{D}^{-\frac{1}{2}}(A + I)\tilde{D}^{-\frac{1}{2}} \tag{9}$$

where $A$ denotes the adjacency matrix, $I$ represents the identity matrix for self-loop addition, and $\tilde{D}$ is the degree matrix of $A + I$. Graph convolution operations at layer $l$ are computed as:

$$\mathbf{H}^{(l)} = \sigma(\hat{A}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)}) \tag{10}$$

where $\mathbf{H}^{(l-1)}$ denotes input features initialized from the enhanced GAT module, $\mathbf{W}^{(l)} \in \mathbb{R}^{d_{l-1} \times d_l}$ represents learnable weights, and $\sigma$ applies RELU activation. The framework employs 2 GCN layers, balancing receptive field expansion

with computational efficiency while avoiding over-smoothing. The GCN component complements GAT's local attention by capturing global graph properties, including community structure, centrality patterns, and long-range dependencies.

### E. BERT-Enhanced Edge Representation

The BERT transformer component generates contextualized edge representations by processing sequences incorporating node embeddings from both transaction endpoints. For each edge $e = (u, v)$, embeddings $\mathbf{h}_u$ and $\mathbf{h}_v$ are extracted from GCN output. Sequences are constructed with special tokens:

$$s_e = [\mathbf{e}_{\text{CLS}}, \mathbf{h}_u, \mathbf{e}_{\text{PAD}}, \mathbf{h}_v, \mathbf{e}_{\text{MASK}}, \mathbf{e}_{\text{SEP}}] \tag{11}$$

where [CLS] and [SEP] denote sequence delimiters, [PAD] handles variable-length padding, and [MASK] enables prediction of missing transaction attributes. The BERT encoder's hidden state is given as:

$$\mathbf{z}_e = \text{BERT}(s_e; \theta_{\text{bert}}) \tag{12}$$

extracting the [CLS] token representation as the edge-level contextualized embedding. This transformer processing captures complex dependencies between transaction attributes that conventional feature engineering cannot represent.

### F. Edge Vulnerability Classification

Final edge representations concatenate sender embeddings, receiver embeddings, and Transformer-enhanced features:

$$\mathbf{h}_e = [\mathbf{h}_u \| \mathbf{h}_v \| \mathbf{z}_e] \tag{13}$$

preserving directional transaction information critical for distinguishing attack patterns. Edge representations are processed through a two-layer MLP classifier with batch normalization (momentum 0.1) and dropout (rate 0.2). Vulnerability predictions are computed via sigmoid activation producing probabilities $\hat{y}_e \in [0, 1]$.

To address the severe class imbalance between legitimate and fraudulent transactions, the framework employs Focal Loss:

$$\mathcal{L}_{FL} = -\alpha_t (1 - p_t)^{\gamma} \log(p_t) \tag{14}$$

where $\gamma = 2.0$ serves as the focusing parameter that down-weights well-classified examples, and $\alpha_t$ balances positive and negative classes. This loss function addresses minority class underweighting during optimization, maintaining detection sensitivity under imbalanced conditions.

## V. Adversarial Attack Scenarios

Adversarial robustness constitutes a critical requirement for deploying intelligent detection systems in production blockchain environments where malicious actors actively attempt to evade detection. This section presents three adversarial attack strategies employed to evaluate HDGN robustness.

### A. Perturbation Edge Attack (PEA)

Perturbation edge attack simulates adversaries who manipulate transaction network structure by adding spurious edges or removing legitimate connections to disguise fraudulent activities. This attack targets graph topology directly, exploiting the dependency of graph neural networks on structural information for neighborhood aggregation. Algorithm 2 formalizes the perturbation procedure.

**Algorithm 2** Perturbation Edge Attack (PEA)

**Require:** Graph $G_t = (V_t, E_t)$, perturbation rate $p_{\text{pert}} = 0.10$
**Ensure:** Perturbed graph $G'_t = (V_t, E'_t)$
Compute $n_{\text{pert}} \leftarrow \lfloor p_{\text{pert}} \times |E_t| \rfloor$
Sample edges to remove: $E_{\text{sample}} \leftarrow \text{RandomSample}(E_t, n_{\text{pert}})$
Generate $E_{\text{new}}$: sample $n_{\text{pert}}$ non-edges uniformly from $\bar{E}_t = (V_t \times V_t) \setminus E_t$
Apply perturbations: $E'_t \leftarrow (E_t \setminus E_{\text{sample}}) \cup E_{\text{new}}$
**return** $G'_t = (V_t, E'_t)$

The attack perturbs 10% of edges in test graphs, representing scenarios where attackers control a substantial portion of the transaction network topology.

### B. Poisoning Attack (POA)

Poisoning attack targets the training phase by injecting mislabeled samples, corrupting learned model parameters, and degrading generalization performance. This attack simulates sophisticated adversaries with access to training data collection or labeling processes. Algorithm 3 describes the label corruption procedure.

**Algorithm 3** Poisoning Attack (POA)

**Require:** Training set $\mathcal{D}_{\text{train}} = \{(e_i, y_i)\}_{i=1}^{N}$, $y_i \in \{0, 1\}$, poisoning rate $p_{\text{poison}} = 0.10$
**Ensure:** Poisoned training set $\mathcal{D}'_{\text{train}}$
Compute $n_{\text{poison}} \leftarrow \lfloor p_{\text{poison}} \times N \rfloor$
Sample indices: $I_{\text{poison}} \leftarrow \text{RandomSample}(\{1, \ldots, N\}, n_{\text{poison}})$
Flip labels: $\mathcal{D}'_{\text{train}} \leftarrow \{(e_i, 1 - y_i) : i \in I_{\text{poison}}\} \cup \{(e_i, y_i) : i \notin I_{\text{poison}}\}$
**return** $\mathcal{D}'_{\text{train}}$

The 10% poisoning rate represents a realistic threat level where adversaries compromise a minority of training labels, sufficient to degrade model performance without triggering data quality monitoring systems.

### C. Temporal Evasion Attack

Temporal evasion attack exploits the time-dependent nature of transaction networks by shifting transaction timestamps to evade temporal pattern detection mechanisms. Adversaries strategically delay or advance transaction execution to disrupt temporal correlations learned during training. Algorithm 4 specifies the timestamp manipulation procedure.

**Algorithm 4** Temporal Evasion Attack

**Require:** Graph $G_t$ with timestamps $\{t_e : e \in E_t\}$, shift $\Delta t_{\text{shift}} = 24$ hours
**Ensure:** Time-shifted graph $G'_t$
Initialize $G'_t \leftarrow G_t$
Shift timestamps: $\forall e \in E'_t : t'_e \leftarrow t_e + \Delta t_{\text{shift}}$
Recompute temporal features with $\{t'_e\}$:
    burstiness $b_v = \sigma_{\Delta t} / \mu_{\Delta t}$
    recency $r_v = t_{\text{now}} - t'_{\text{last}}$
    frequency $f_v$
**return** $G'_t$

The 24-hour shift represents a full diurnal cycle, disrupting daily activity patterns while maintaining transaction ordering. Timestamp shifting may cause edges to migrate between temporal snapshots; snapshot membership is recomputed after shifting. For evaluation, PEA and temporal evasion are applied at test time, while POA corrupts training data with clean validation and test sets.

## VI. Experimental Setup

### A. Dataset Description

The experimental evaluation employs four publicly available blockchain transaction network datasets spanning diverse cryptocurrency ecosystems and token standards.

Bitcoin OTC (BOTC) originates from the Bitcoin OTC trading platform, representing trust-weighted transactions among cryptocurrency traders [21]. Bitcoin Alpha (BTA) captures analogous trust-based transaction relationships within an alternative trading community [22]. ERC20 Transaction encompasses fungible token transfer records adhering to the ERC20 standard on Ethereum, extracted from Etherscan covering blocks 0 through 6,999,999 [23]. ERC721 Transaction captures non-fungible token (NFT) transfers compliant with the ERC721 specification, reflecting the digital collectibles ecosystem with distinct transaction patterns.

### B. Baselines Methods

We compare HDGN against five state-of-the-art methods: GCCAD [15] employing graph contrastive learning for anomaly detection; PDGNN [24] using Chebyshev spectral convolutions for phishing detection; CTGN [25] preserving local and global structural similarity through k-core decomposition; STA-GT [19] integrating spatial-temporal awareness with graph transformers; and MGGPT [26] leveraging GPT-enhanced multi-graph representations for dynamic fraud detection.

### C. Hardware and Software Configuration

Experiments were executed on high-performance infrastructure comprising 4× NVIDIA RTX 3090 GPUs (24GB each), Intel Xeon Silver 4314 CPU (64 cores) at 2.40GHz, and 384GB DDR4 RAM. The HDGN framework was implemented using PyTorch 1.12 and PyTorch Geometric 2.1. The Transformer encoder uses 4 layers with 256-dimensional hidden states and 8 attention heads, initialized randomly.

Key hyperparameters include: learning rate $\eta = 0.001$ with Adam optimizer, GAT attention heads $K = 4$, hidden dimension $d = 256$, missing information weight $\lambda = 0.3$, Focal Loss focusing parameter $\gamma = 2.0$, ego network radius $k = 2$, dropout rate 0.3, and training epochs 120 with early stopping patience of 15 epochs. The dataset is split temporally: the first 70% of snapshots for training, the next 15% for validation, and the final 15% for testing. Within each split, stratified sampling ensures consistent class distributions for batch construction. Batch size is 512, random seed is fixed at 42, classification threshold is 0.5, and early stopping monitors validation F1-score.

## VII. Results and Analysis

### A. Comparative Analysis

Table I presents a comprehensive performance comparison against baseline methods across all four datasets.

TABLE I
PERFORMANCE COMPARISON AGAINST FIVE BASELINE METHODS ON FOUR BLOCKCHAIN DATASETS.

| Dataset | Metric | GCCAD | PDGNN | CTGN | STA-GT | MGGPT | HDGN |
|---|---|---|---|---|---|---|---|
| BOTC | Accuracy | 71.9 | 75.1 | 77.1 | 77.3 | 89.2 | **90.4** |
| | AUC | 73.4 | 76.8 | 75.9 | 79.5 | 92.7 | **93.4** |
| | Precision | 57.5 | 72.6 | 73.3 | 77.2 | 94.6 | **95.3** |
| | Recall | 51.9 | 63.5 | 47.6 | 83.3 | 89.9 | **93.2** |
| | F1-score | 54.6 | 67.8 | 58.2 | 80.1 | 92.2 | **94.2** |
| ERC20 | Accuracy | 73.2 | 76.5 | 75.8 | 80.5 | 91.2 | **91.9** |
| | AUC | 72.5 | 75.8 | 74.3 | 81.2 | 93.5 | **94.8** |
| | Precision | 76.2 | 79.5 | 78.8 | 83.5 | 94.8 | **95.7** |
| | Recall | 58.3 | 67.2 | 55.8 | 85.1 | 91.2 | **94.7** |
| | F1-score | 66.1 | 72.9 | 65.3 | 84.3 | 93.0 | **95.2** |
| ERC721 | Accuracy | 70.3 | 73.8 | 75.4 | 78.6 | 92.7 | **94.0** |
| | AUC | 67.3 | 71.5 | 66.5 | 79.6 | 93.4 | **94.5** |
| | Precision | 71.3 | 74.6 | 77.7 | 81.8 | 95.1 | **95.8** |
| | Recall | 55.6 | 66.6 | 54.2 | 78.9 | 85.7 | **91.5** |
| | F1-score | 62.5 | 70.4 | 63.7 | 80.3 | 90.2 | **93.6** |
| BTA | Accuracy | 74.5 | 82.3 | 79.8 | 86.2 | 96.5 | **97.7** |
| | AUC | 69.8 | 67.5 | 71.3 | 80.8 | 89.3 | **96.2** |
| | Precision | 56.5 | 75.6 | 83.1 | 85.6 | 95.8 | **96.5** |
| | Recall | 52.8 | 68.7 | 53.7 | 70.5 | 100.0 | **100.0** |
| | F1-score | 54.6 | 72.0 | 65.2 | 77.4 | 97.9 | **98.2** |

All metrics in %. Best results in **bold**.

HDGN achieves the highest measured performance across all five metrics and four datasets, as detailed in Table I. On BOTC, HDGN demonstrates 93.4% AUC, outperforming MGGPT by 0.7 percentage points ($p < 0.05$, paired t-test) and exceeding STA-GT by 13.9 percentage points. The recall of 93.2% exceeds MGGPT by 3.3 percentage points, addressing a specific limitation of prompt-based approaches that favor precision at the expense of detection sensitivity.

On ERC20, representing the most challenging class imbalance scenario (1:8.5 ratio), HDGN achieves 91.9% accuracy, 94.8% AUC, and 95.2% F1-score. The 94.7% recall under severe imbalance represents a 3.5 percentage point improvement over MGGPT, demonstrating that the combination of Focal Loss, missing information prediction, and hybrid architecture effectively mitigates minority class underrepresentation.

On ERC721, HDGN achieves 94.0% accuracy and 93.6% F1-score, with 5.8 percentage points recall enhancement over MGGPT. Against traditional GNN methods (GCCAD, PDGNN), HDGN demonstrates 23.2–31.1 percentage points F1-score gains, validating the hybrid architecture's advantages.

On BTA with the mildest class imbalance (1:1.7), HDGN achieves 97.7% accuracy and 100.0% recall, correctly identifying all vulnerable edges. The 6.9 percentage points AUC improvement over MGGPT indicates superior ranking quality for prioritizing suspicious transactions.

Performance improvements correlate with four architectural components: the missing information prediction module addresses sender-receiver asymmetry, degrading conventional GNN aggregation; the hybrid GAT-GCN pipeline captures

both local attention-weighted patterns and global spectral properties; the BERT-enhanced edge representations model complex attribute dependencies; and Focal Loss explicitly addresses class imbalance during optimization.

### B. Performance Visualization

Figure 2 aggregates performance across all datasets, enabling cross-dataset comparison.

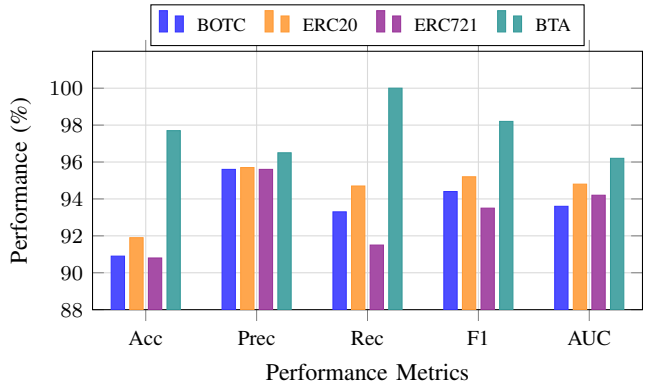


Fig. 2. Edge classification performance comparison across four blockchain datasets.

BTA achieves the highest performance (97.7% accuracy), attributed to its lower class imbalance ratio. ERC20 maintains strong performance (91.9% accuracy) despite the most severe imbalance, validating the effectiveness of Focal Loss optimization and the missing information prediction module. BOTC and ERC721 maintain consistent 90.4–94.0% accuracy ranges despite varying network scales and class distributions.

The limited performance variance (7.3 percentage points accuracy range) across datasets with vastly different characteristics validates the architecture's generalization capability. From a practical deployment perspective, consistently high precision (95.8% average) minimizes false alarm rates that would overwhelm security analysts, while strong recall (94.8% average) ensures detection of the majority of genuine threats. This balance suggests potential applicability to blockchain security monitoring, pending validation of computational scalability.

### C. Ablation Study

To quantify individual component contributions, we conducted systematic ablation experiments on the BOTC dataset. Table II presents the results.

TABLE II
ABLATION STUDY RESULTS ON THE BOTC DATASET

| Configuration | Acc | AUC | Prec | Rec | F1 |
|---|---|---|---|---|---|
| Full HDGN | **90.4** | **93.4** | **95.3** | **93.2** | **94.2** |
| W/o GAT | 86.7 | 89.8 | 91.5 | 89.4 | 90.4 |
| W/o Missing Info | 87.9 | 91.2 | 93.1 | 90.6 | 91.8 |
| W/o GCN | 88.2 | 91.5 | 93.4 | 91.1 | 92.2 |
| W/o BERT | 87.5 | 90.8 | 92.8 | 90.2 | 91.5 |
| W/o Focal Loss | 88.6 | 91.9 | 94.2 | 89.8 | 91.9 |

W/o: Without. All metrics in %. Best results in **bold**.

Removing GAT produces the largest degradation ($-3.7$pp accuracy, $-3.8$pp F1), confirming adaptive attention as the most critical contribution. BERT removal yields the second-largest impact ($-2.9$pp accuracy), while the missing information module contributes 2.5pp improvement. Replacing Focal Loss primarily impacts recall ($-3.4$pp), confirming its role in class imbalance handling. All components contribute meaningfully to overall performance.

### D. Adversarial Robustness Results

Figure 3 quantifies performance degradation under three adversarial scenarios.

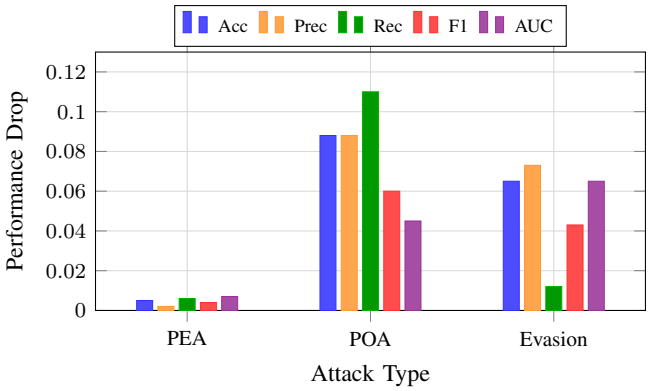


Fig. 3. Performance degradation under adversarial attacks. PEA shows exceptional robustness (<1pp drops), POA exhibits moderate vulnerability (4.5–11.0pp), and Evasion produces moderate degradation (1.2–7.3pp).

Under perturbation edge attack (PEA), HDGN exhibits robustness with 0.8pp accuracy degradation and 0.6pp F1-score degradation, validating that attention mechanisms distinguish legitimate patterns from adversarial manipulation.

Poisoning attack (POA) represents the primary challenge, with 8.8pp accuracy and 11.0pp recall drops. This vulnerability identifies training data integrity as critical for deployment.

Temporal evasion produces moderate degradation (6.5pp accuracy) with minimal recall impact (1.2pp), revealing that structural patterns serve as primary indicators rather than temporal features. Organizations should prioritize training data quality to mitigate POA vulnerability.

## VIII. Conclusion and Future Work

This paper presented hybrid dynamic graph networks (HDGN), an intelligent system for robust vulnerability detection in blockchain transaction networks. The framework addresses critical limitations of existing approaches through four principal contributions: a hybrid GAT-GCN-BERT architecture capturing multi-scale transaction patterns, a missing information prediction module addressing sender-receiver structural asymmetry, a dual vulnerability scoring scheme enabling a four-class behavioral pattern taxonomy, and comprehensive adversarial robustness evaluation under perturbation, poisoning, and evasion attacks.

Extensive experiments across four blockchain datasets demonstrated HDGN's effectiveness, achieving higher average

accuracy, F1-score, and AUC while consistently outperforming five state-of-the-art baselines. The framework exhibited strong resilience to structural perturbation attacks with less than one percentage point degradation, and ablation studies confirmed meaningful contributions from each architectural component.

Three limitations constrain the current implementation: scalability constraints for million-node production networks, computational overhead from BERT components limiting real-time applicability, and sensitivity to training label corruption under poisoning attacks.

Future research directions encompass developing robust loss functions to mitigate poisoning vulnerability, incorporating online learning mechanisms for streaming transaction scenarios, and enhancing model explainability for forensic analysis applications in blockchain security investigations.

## ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China (No. U22B2029).

## REFERENCES

[1] D. Cheng, Y. Zou, S. Xiang, and C. Jiang, "Graph neural networks for financial fraud detection: a review," *Frontiers of Computer Science*, vol. 19, no. 9, p. 199609, Jan. 2025, doi: 10.1007/s11704-024-40474-y.

[2] Q. Xia, B. Ansu, J. Gao, G. M. Ntuala, H. Xia, and O. I. Amankona, "A novel time series approach to anomaly detection and correction for complex blockchain transaction networks," in *Proc. IEEE 23rd Int. Conf. Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2024, pp. 674–682, doi: 10.1109/TrustCom63139.2024.00106.

[3] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, arXiv:1710.10903, doi: 10.48550/arXiv.1710.10903.

[4] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. B. Schardl, and C. E. Leiserson, "EvolveGCN: Evolving graph convolutional networks for dynamic graphs," in *Proc. AAAI Conf. Artificial Intelligence*, vol. 34, no. 4, 2020, pp. 5363–5370, doi: 10.48550/arXiv.1902.10191.

[5] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2020, doi: 10.48550/arXiv.2002.07962.

[6] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, 2020, pp. 66–74, doi: 10.48550/arXiv.2005.10203.

[7] B. Xu, Y. Wang, X. Liao, and K. Wang, "Efficient fraud detection using deep boosting decision trees," *Decision Support Systems*, vol. 175, p. 114037, 2023, doi: 10.1016/j.dss.2023.114037.

[8] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: a survey," 2019, arXiv:1901.03407, doi: 10.48550/arXiv.1901.03407.

[9] I. Alarab, S. Prakoonwit, and M. I. Nacer, "Competence of graph convolutional networks for anti-money laundering in Bitcoin blockchain," in *Proc. 5th Int. Conf. Machine Learning Technologies (ICMLT)*, 2020, pp. 23–27, doi: 10.1145/3409073.3409080.

[10] A. Asiri and K. Somasundaram, "Graph convolution network for fraud detection in bitcoin transactions," *Sci. Rep.*, vol. 15, no. 1, p. 11076, 2025, doi: 10.1038/s41598-025-95672-w.

[11] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Information Processing Systems (NeurIPS)*, 2017, pp. 1025–1035, doi: 10.5555/3294771.3294869.

[12] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *Proc. World Wide Web Conf. (WWW)*, 2019, pp. 2022–2032, doi: 10.1145/3308558.3313562.

[13] T. Bilot, G. Geis, and B. Hammi, "PhishGNN: A phishing website detection framework using graph neural networks," in *Proc. 19th Int. Conf. Security and Cryptography (SECRYPT)*, 2022, pp. 424–431, doi: 10.5220/0011328600003283.

[14] D. Cheng, X. Wang, Y. Zhang, and L. Zhang, "Graph neural network for fraud detection via spatial-temporal attention," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3800–3813, 2022, doi: 10.1109/TKDE.2020.3025588.

[15] B. Chen, J. Zhang, X. Zhang, Y. Dong, J. Song, P. Zhang, K. Xu, E. Kharlamov, and J. Tang, "GCCAD: Graph contrastive coding for anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 8, pp. 8037–8051, 2023, doi: 10.1109/TKDE.2022.3200459.

[16] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, "Temporal graph networks for deep learning on dynamic graphs," 2020, arXiv:2006.10637, doi: 10.48550/arXiv.2006.10637.

[17] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, "DySAT: Deep neural representation learning on dynamic graphs via self-attention networks," in *Proc. 13th ACM Int. Conf. Web Search and Data Mining (WSDM)*, 2020, pp. 519–527, doi: 10.1145/3336191.3371845.

[18] S. Chen, Y. Liu, Q. Zhang, Z. Shao, and Z. Wang, "Multi-distance spatial-temporal graph neural network for anomaly detection in blockchain transactions," *Adv. Intell. Syst.*, vol. 7, no. 8, p. 2400898, 2025, doi: 10.1002/aisy.202400898.

[19] Y. Tian and G. Liu, "Spatial-temporal-aware graph transformer for transaction fraud detection," *IEEE Trans. Ind. Informat.*, vol. 20, no. 11, pp. 12659–12668, 2024, doi: 10.1109/TII.2024.3423447.

[20] Z. Liu, T.-K. Nguyen, and Y. Fang, "Tail-GNN: Tail-node graph neural networks," in *Proc. 27th ACM SIGKDD Conf. Knowledge Discovery & Data Mining*, 2021, pp. 1109–1119, doi: 10.1145/3447548.3467276.

[21] S. Kumar, F. Spezzano, V. S. Subrahmanian, and C. Faloutsos, "Edge weight prediction in weighted signed networks," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, 2016, pp. 221–230, doi: 10.1109/ICDM.2016.0033.

[22] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. S. Subrahmanian, "REV2: Fraudulent user prediction in rating platforms," in *Proc. 11th ACM Int. Conf. Web Search and Data Mining (WSDM)*, 2018, pp. 333–341, doi: 10.1145/3159652.3159729.

[23] P. Zheng, Z. Zheng, J. Wu, and H.-N. Dai, "XBlock-ETH: Extracting and exploring blockchain data from Ethereum," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 95–106, 2020, doi: 10.1109/OJCS.2020.2990458.

[24] P. Li, Y. Xie, X. Xu, J. Zhou, and Q. Xuan, "Phishing fraud detection on Ethereum using graph neural network," 2022, arXiv:2204.08194, doi: 10.48550/arXiv.2204.08194.

[25] J. Liu, C. Xu, C. Yin, W. Wu, and Y. Song, "K-core based temporal graph convolutional network for dynamic graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3841–3853, 2022, doi: 10.1109/TKDE.2020.3033829.

[26] A. Badjie, G. M. Ntuala, Q. Xia, J. Gao, and H. Xia, "MGGPT: A multi-graph GPT-enhanced framework for dynamic fraud detection in cryptocurrency networks," *Computer Networks*, vol. 270, p. 111508, 2025, doi: 10.1016/j.comnet.2025.111508.